IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.15328

AWS BATCH FOR BATCH PROCESSING OF DOCKER CONTAINER

P. Kishan [1], Dr . S. Thavamani [2]

III BCA, Department of Computer Applications,

Sri Ramakrishna College of Arts & Science (Autonomous), Coimbatore, Tamil Nadu, India[1]

Associate Professor, Department of Computer Applications,

Sri Ramakrishna College of Arts & Science (Autonomous), Coimbatore, Tamil Nadu, India[2]

Abstract: Cloud computing has significantly transformed the way large-scale computational workloads are processed and managed. Traditional systems often struggle to efficiently handle highvolume batch jobs due to limitations in scalability, resource allocation, and automation. This paper presents a scalable solution for batch processing using **AWS Batch** integrated with containerized applications built using **Docker**.

The proposed system enables developers to run large numbers of batch computing jobs without manually provisioning or managing infrastructure. AWS Batch dynamically provisions the optimal quantity and type of compute resources based on the volume and requirements of submitted jobs. By packaging applications and their dependencies into Docker containers, the system ensures consistent execution environments, improved portability, and simplified deployment across cloud infrastructures.

In this project, Docker containers are stored and managed through **Amazon Elastic Container Registry (ECR)** and executed on scalable compute resources such as **Amazon EC2** instances. AWS Batch automatically handles job scheduling, queue management, resource allocation, and monitoring, enabling efficient batch execution for workloads such as data processing, scientific simulations, and large-scale analytics.

The implemented architecture demonstrates how containerized workloads can be efficiently orchestrated using AWS Batch to improve performance, reduce operational complexity, and optimize resource utilization. The system also supports automated scaling and fault-tolerant execution, ensuring reliable job completion even under heavy workloads.

Overall, this research highlights the advantages of integrating AWS Batch with Docker-based containerization to build a robust, scalable, and cost-effective batch processing framework suitable for modern cloud-based applications.

I. INTRODUCTION

Cloud computing has become one of the most important technologies for handling large-scale data processing and complex computational workloads in modern information systems. Many organizations, research institutions, and industries require powerful computing environments to process huge amounts of data efficiently. Traditional computing infrastructures often face limitations such as high hardware costs, lack of scalability, and complex resource management. In such systems, administrators must manually configure servers, allocate computing resources, and manage workloads, which can lead to inefficiency and delays. To overcome these challenges, cloud-based platforms provide scalable, flexible, and cost-effective solutions that allow users to run computational workloads without maintaining physical infrastructure. One of the most effective approaches for managing large computational tasks in cloud environments is batch processing.

Batch processing refers to the execution of a group of tasks or jobs without manual intervention, where tasks are processed sequentially or simultaneously in an automated manner. It is widely used in applications such as data analysis, scientific simulations, financial processing, video rendering, and machine learning workloads. However, managing batch jobs in traditional environments can be complex due to the need for scheduling systems, resource allocation mechanisms, and monitoring tools. Cloud technologies simplify this process by providing managed services that automate these operations.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026
DOI: 10.17148/IJARCCE.2026.15328

AWS Batch is a fully managed cloud service provided by **Amazon Web Services** that enables developers, engineers, and researchers to efficiently run hundreds or thousands of batch computing jobs on the cloud. The service automatically provisions the required compute resources, schedules jobs based on priority, and optimizes resource utilization without requiring users to manually configure infrastructure. By using AWS Batch, organizations can focus on application development while the service handles job queues, scaling, and resource management. This significantly reduces operational complexity and improves overall system efficiency.

Another important technology that plays a key role in modern cloud computing is containerization. Containerization allows applications and their dependencies to be packaged into lightweight, portable units called containers. These containers ensure that applications run consistently across different computing environments. **Docker** is one of the most widely used containerization platforms that enables developers to build, package, and deploy applications in isolated environments. Docker containers include all necessary libraries, runtime components, and system tools required to run the application, which eliminates compatibility issues between development and production environments.

In cloud-based batch processing systems, Docker containers provide a reliable way to execute applications while maintaining consistency and portability. Container images can be stored and managed in repositories such as **Amazon Elastic Container Registry**, which allows developers to securely store and retrieve container images when needed. These containerized applications can then be executed on scalable computing resources such as **Amazon Elastic Compute Cloud** instances. By combining container technology with cloud computing services, organizations can create highly efficient and scalable batch processing systems.

The integration of AWS Batch with Docker containers provides a powerful framework for processing large volumes of jobs in an automated and scalable environment. AWS Batch manages job queues, determines optimal compute resources, and launches containerized workloads automatically. This approach eliminates the need for manual server provisioning and reduces the risk of resource underutilization. Additionally, the system can dynamically scale resources depending on the workload demand, ensuring that jobs are processed efficiently even during peak workloads.

In this project, a batch processing system is designed using AWS Batch and Docker containers to demonstrate how containerized workloads can be efficiently executed in a cloud environment. The system architecture includes container image creation using Docker, storage of images in Amazon Elastic Container Registry, job submission through AWS Batch job queues, and execution on Amazon EC2 compute environments. This architecture ensures efficient workload management, improved reliability, and better resource utilization.

II. RELATED WORK

In recent years, cloud computing has become a major platform for executing large-scale data processing and computational workloads. Many researchers and organizations have explored different approaches to improve batch processing performance, scalability, and resource utilization in cloud environments. Traditional batch processing systems relied on cluster-based infrastructures where administrators manually configured computing resources and job scheduling systems. While these systems were capable of executing large workloads, they required significant infrastructure management and often lacked dynamic scalability.

With the emergence of cloud computing services provided by **Amazon Web Services**, several solutions were introduced to simplify large-scale workload execution. One of the earliest approaches involved using virtual machines through **Amazon Elastic Compute Cloud** to create custom batch processing environments. In this approach, developers manually deployed applications on virtual machines and used custom scripts or scheduling tools to manage job execution. Although this method provided flexibility, it required manual provisioning, monitoring, and scaling of computing resources, which increased operational complexity.

To address these challenges, containerization technology emerged as a powerful solution for application deployment and workload management. **Docker** introduced lightweight containers that allow applications and their dependencies to be packaged together and executed consistently across different environments. Researchers have widely adopted Docker containers in distributed computing systems because they simplify deployment, improve portability, and reduce configuration issues. Container-based architectures also enable faster startup times compared to traditional virtual machine-based systems.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026
DOI: 10.17148/IJARCCE.2026.15328

In addition to containerization, orchestration tools have been developed to manage large numbers of containers in cloud environments. Platforms such as **Kubernetes** provide automated container orchestration, resource scheduling, and load balancing capabilities. Kubernetes-based systems allow developers to deploy and manage containerized applications at scale, making them suitable for microservices and distributed computing environments. However, configuring and maintaining such orchestration platforms may still require specialized knowledge and operational effort.

The research presented in this project builds upon these existing technologies by integrating AWS Batch with Docker-based container environments for efficient batch job execution. The proposed system leverages automated resource provisioning, scalable compute environments, and containerized workloads to improve performance and reduce operational overhead. Compared to traditional batch processing systems, this approach offers greater flexibility, easier deployment, and better scalability for handling large volumes of computational tasks.

Overall, previous research and technological advancements in cloud computing, containerization, and automated workload management have laid the foundation for modern batch processing solutions. The integration of AWS Batch with Docker containers represents an effective and practical approach for building scalable batch processing systems in cloud environments.

III. OBJECTIVES AND CHALLENGES

The main aim of this project is to develop an efficient and scalable batch processing system using **AWS Batch** with containerized applications created using **Docker**. Cloud-based batch processing helps organizations execute large computational workloads without maintaining complex infrastructure. By integrating container technology with cloud services, the system can automatically manage resources, schedule jobs, and execute tasks efficiently. The proposed system focuses on improving scalability, automation, and reliability in batch job execution while reducing operational complexity. Container images are securely stored using **Amazon Elastic Container Registry** and executed on scalable compute environments such as **Amazon Elastic Compute Cloud** instances.

Objectives

1. To design and implement a scalable batch processing system using **AWS Batch**.
2. To automate job scheduling and execution without manual infrastructure management.
3. To utilize **Docker** containers for consistent and portable application deployment.
4. To improve resource utilization through automatic scaling of compute resources.
5. To ensure reliable and fault-tolerant execution of batch workloads.
6. To reduce operational complexity and infrastructure management overhead.
7. To enable secure storage and retrieval of container images using **Amazon Elastic Container Registry**.
8. To support efficient processing of large-scale computational jobs in cloud environments.

Challenges

1. Efficient allocation and management of cloud computing resources.
2. Managing large numbers of batch jobs and scheduling them effectively.
3. Ensuring security and access control in cloud-based environments.
4. Handling failures and ensuring reliable job execution.
5. Monitoring and debugging distributed batch workloads.
6. Managing container images and maintaining version control.
7. Reducing latency and improving performance when processing large datasets.
8. Controlling operational costs while maintaining system scalability.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.15328

IV. SYSTEM ARCHITECTURE

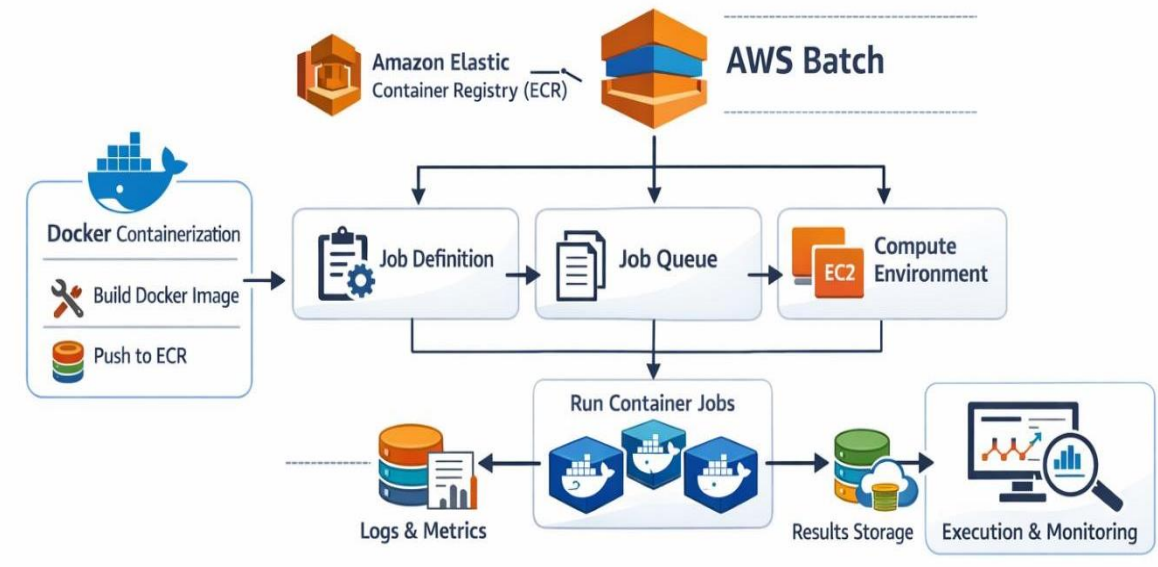
The system architecture of the proposed batch processing framework is designed to efficiently execute large-scale workloads using cloud-based infrastructure. The architecture integrates **AWS Batch** with containerized applications developed using **Docker**, enabling automated job scheduling, resource allocation, and scalable execution of batch tasks. The overall architecture consists of several key components including job definition, job queues, compute environments, container image repositories, and cloud-based compute resources. These components work together to ensure efficient processing of batch workloads while maintaining scalability and reliability.

In the proposed architecture, applications are first containerized using Docker. Containerization allows the application along with its dependencies, libraries, and runtime environment to be packaged into a single container image. This ensures that the application runs consistently across different computing environments without compatibility issues. Once the container image is created, it is securely stored in **Amazon Elastic Container Registry**, which acts as a centralized repository for managing container images. The container image stored in the repository can then be accessed by AWS Batch whenever a job needs to be executed.

The job execution process begins with the creation of a job definition in AWS Batch. A job definition specifies the container image, resource requirements such as CPU and memory, and other runtime parameters needed for executing the batch job. After defining the job configuration, jobs are submitted to a job queue where they wait until the required compute resources become available. AWS Batch automatically evaluates the job queue and determines the optimal order in which jobs should be executed based on priority and resource availability.

The compute environment is another critical component of the architecture. It defines the type and quantity of computing resources required for executing batch jobs. AWS Batch automatically provisions and manages compute resources such as **Amazon Elastic Compute Cloud** instances according to the workload demand. When jobs are submitted, AWS Batch dynamically launches the required compute resources and deploys the corresponding Docker containers to execute the tasks. This automatic scaling capability ensures that the system can handle varying workloads efficiently while minimizing idle resource usage.

Once the compute resources are available, the Docker container is pulled from the container registry and executed within the compute environment. The batch job runs inside the container and processes the assigned workload. After the job is completed, the output results can be stored in cloud storage services or databases for further analysis. AWS Batch also provides monitoring and logging features that allow developers to track job execution status, analyze performance metrics, and troubleshoot potential issues.



IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.15328

V. IMPLEMENTATION

The implementation of the system is carried out using **AWS Batch** and container technology through **Docker** to process batch workloads efficiently. First, the application and its required dependencies are packaged into a Docker container image. This container image is then stored securely in **Amazon Elastic Container Registry**, which acts as a repository for managing container images.

After storing the container image, a job definition is created in AWS Batch specifying the container image, memory, and CPU requirements. Jobs are then submitted to a job queue where they wait until resources become available. AWS Batch automatically provisions compute resources using **Amazon Elastic Compute Cloud** instances and executes the containerized jobs. The system monitors job execution and records logs for tracking performance and identifying failures. Once the job execution is completed, the output data is generated and stored for further analysis. This implementation ensures efficient, scalable, and automated batch job processing in a cloud environment.

VI. EVALUATION RESULTS AND DISCUSSIONS

The performance of the proposed batch processing system was evaluated using cloud infrastructure services provided by **Amazon Web Services**. The system was implemented using **AWS Batch** with containerized applications created through **Docker**. The evaluation focused on analyzing system performance, scalability, job execution efficiency, and resource utilization while processing batch workloads in the cloud environment.

During the testing phase, multiple batch jobs were submitted to the AWS Batch job queue. The system successfully scheduled and executed the jobs by automatically provisioning compute resources using **Amazon Elastic Compute Cloud** instances. The container images required for job execution were retrieved from **Amazon Elastic Container Registry**, ensuring consistent deployment of applications across the compute environment. The results showed that the system effectively handled multiple job requests while maintaining stable performance and reliable execution.

The evaluation results demonstrate that the proposed system provides efficient batch job processing by automatically scaling computing resources based on workload demand. When the number of jobs increased, additional compute instances were launched to handle the workload, ensuring minimal waiting time in the job queue. Similarly, when the workload decreased, unused resources were automatically terminated, reducing operational costs and improving resource efficiency.

Monitoring tools and log analysis were used to observe system behavior during execution. The results indicated that the containerized environment ensured consistent job execution without configuration errors. The automated scheduling and resource allocation capabilities of AWS Batch significantly reduced manual intervention and improved overall system performance.

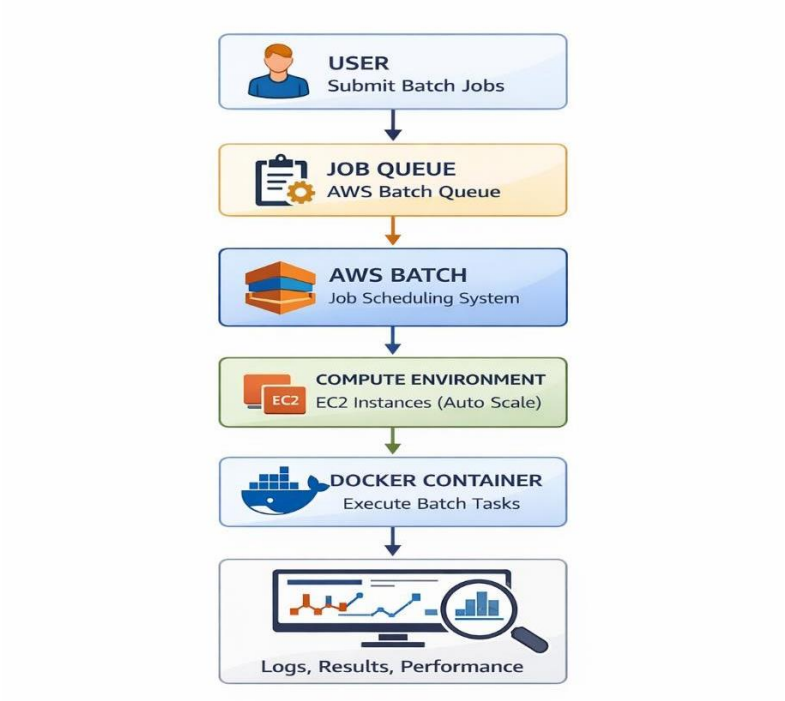
From the experimental results, it is evident that the integration of AWS Batch with Docker containers provides a reliable and scalable solution for batch processing workloads. The system successfully demonstrates improved efficiency, reduced infrastructure management effort, and better resource optimization compared to traditional batch processing methods. These findings confirm that cloud-based containerized batch processing systems are highly suitable for largescale computational workloads in modern cloud environments.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.15328



VII. CONCLUSION

The proposed batch processing system using **AWS Batch** and **Docker** demonstrates an efficient and scalable approach for executing large computational workloads in a cloud environment. By utilizing containerization and automated resource management, the system simplifies job scheduling, improves resource utilization, and reduces manual infrastructure management. The integration with **Amazon Elastic Compute Cloud** allows dynamic scaling of compute resources based on workload demand, ensuring efficient processing of batch jobs.

Furthermore, the system provides improved reliability and consistency because containerized applications run in a controlled environment with all required dependencies. The automated scheduling and monitoring features help reduce operational complexity and improve system performance. The evaluation results show that the proposed architecture can handle multiple batch jobs effectively while maintaining stability. In addition, the use of cloud infrastructure helps minimize hardware maintenance costs and enhances flexibility for future expansion. Overall, the implementation proves that cloud-based batch processing systems are highly suitable for handling modern large-scale data processing tasks

VIII. FUTURE ENHANCEMENTS

Future improvements to the proposed system can further enhance performance, scalability, and automation. The batch processing framework built using **AWS Batch** and **Docker** can be extended by integrating additional cloud services and advanced monitoring tools. These enhancements will help improve system efficiency and provide better management of largescale workloads.

In the future, the system can incorporate advanced monitoring and analytics tools to analyze job performance and resource utilization more effectively. Integration with **Amazon CloudWatch** can provide detailed insights into system performance and enable real-time alerts for failures or performance issues. Additionally, support for multiple compute environments using **Amazon Elastic Compute Cloud** and serverless technologies can further improve scalability and reduce operational costs.

Another possible enhancement is the integration of machine learning techniques for intelligent job scheduling and workload prediction. This can help optimize resource allocation and reduce processing time for large datasets. Security improvements such as advanced access control, encryption mechanisms, and container vulnerability scanning can also be implemented to strengthen system security.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026
DOI: 10.17148/IJARCCE.2026.15328

Overall, these future enhancements will make the batch processing system more intelligent, secure, and efficient, enabling it to handle more complex workloads and support large-scale cloud-based applications in the future.

REFERENCES

[1]. Amazon Web Services, *AWS Batch User Guide*, Amazon Web Services Documentation, 2024.
[2]. Docker Documentation, *Docker Containerization Platform*, Docker Inc., 2024.
[3]. Amazon Elastic Compute Cloud Documentation, Amazon Web Services, 2024.
[4]. Amazon Elastic Container Registry Documentation, Amazon Web Services, 2024.
[5]. Amazon CloudWatch Documentation, Amazon Web Services, 2024.
[6]. Merkel, D., *Docker: Lightweight Linux Containers for Consistent Development and Deployment*, Journal of Linux Technology, 2014.
[7]. Pahl, C., *Containerization and the PaaS Cloud*, IEEE Cloud Computing Journal, 2015.